AD-A035 156 CONTROL DATA CORP MINNEAPOLIS MINN DIGITAL IMAGE SYS--ETC F/G 8/2
DIGITAL CARTOGRAPHIC STUDY AND BENCHMARK.(U)
DEC 75 D J PANTON, M E MURPHY DAAG53-75-C-0195

UNCLASSIFIED ETL-0091 NL

1 OF 1
ADA035156

END

DATE FILMED
3 - 77

(18) ETL-0091 (19)

(2)

(6) DIGITAL CARTOGRAPHIC STUDY AND BENCHMARK.

SECOND INTERIM TECHNICAL REPORT

(9) Interim technical rept. no. 2,

Prepared for

U. S. Army Engineer Topographic Laboratories
For Belvoir, Virginia

(15) Contract DAAG53-75-C-0195

Prepared by
(10) D. J. Panton
M. E. Murphy

Approved For Public Release
Distribution Unlimited

DIGITAL IMAGE SYSTEMS DIVISION
Control Data Corporation
Minneapolis, Minnesota

(11) December 1975 (12) 32 p.

408 732 dn

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER ETL-0091	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) DIGITAL CARTOGRAPHIC STUDY AND BENCHMARK SECOND INTERIM TECHNICAL REPORT		5. TYPE OF REPORT & PERIOD COVERED Contract 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) D.J. Panton M.E. Murphy		8. CONTRACT OR GRANT NUMBER(s) DAAG53-75-C-0195
9. PERFORMING ORGANIZATION NAME AND ADDRESS Control Data Corporation 2800 East Old Shakopee Road Minneapolis, Minnesota 55440		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS U.S. Army Engineer Topographic Laboratories Fort Belvoir, Virginia		12. REPORT DATE December 1975 13. NUMBER OF PAGES 28
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for Public Release; Distribution Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Stereo Image Matching	Microprocessors
Altorithm Distribution	Match Point Reliability
Block Matching Algorithm	Correlation Patch Shaping
Microprogrammable Processors	Flexible Processor(FP) Array

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report describes the practicality of implementing the automatic process of stereo image matching on a configuration of extremely fast micro-programmable processors. The report includes a three-phase program designed to bridge the gap between mathematical algorithm development and an actual hardware benchmark realization. The report concludes with the block matching algorithm being logically reconstructed and ready for microprogramming.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

DIGITAL CARTOGRAPHIC STUDY AND BENCHMARK
SECOND INTERIM TECHNICAL REPORT

Prepared for

U. S. Army Engineer Topographic Laboratories
Fort Belvoir, Virginia

Contract DAAG53-75-C-0195

Prepared by

D. J. Panton
M. E. Murphy

DIGITAL IMAGE SYSTEMS DIVISION
Control Data Corporation
Minneapolis, Minnesota

December 1975

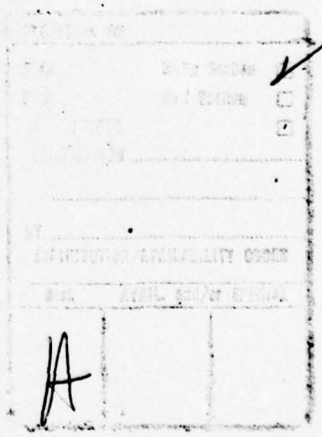


TABLE OF CONTENTS

LIST OF FIGURES

1.0 INTRODUCTION

This technical report is a summary of the work performed during the second phase of an ongoing three-phase effort. The purpose of the effort is to assess the practicality of implementing the automatic process of stereo image matching on a configuration of extremely fast microprogrammable processors.

In general, the three-phase program is designed to provide the researcher with a means for bridging the gap between mathematical algorithm development and an actual hardware benchmark realization. The steps of this generalized program are illustrated in Figure 1-1.

In the present context of a stereo image matching benchmark both Phase A and Phase B have been completed. The objective of Phase A was to redesign and modify in a general purpose software environment two existing image matching approaches, the strip approach and the block approach. As a result of Phase A, it was determined that the block approach best suited the requirements of and the flexibility necessary for digital stereo mapping [2].

The goal then of Phase B was to break the block process apart into its logical components in such a way that its implementation in microcode is straightforward. The justification for this lies in the fact that algorithms that have been developed in a higher level programming language and that run on general purpose sequential machines are generally too slow to be practical in a large data volume image processing environment. What is practical is the incorporation of the desired algorithm in a hardwired machine that is capable of the utmost process parallelism and processing speed. However, this method of implementation is rather inflexible as regards algorithm modification and machine versatility. The availability of arrays of microprocessors at first seems to offer the researcher the best of both worlds; the speed of microprocessors can approach that of hardwired machines and flexibility is maintained in that the processors are microprogrammable.

However, microprogramming is performed in a data flow and machine logic control language whereas the algorithm was developed in a mathematical, problem-oriented language. The direct transliteration of the higher level language into

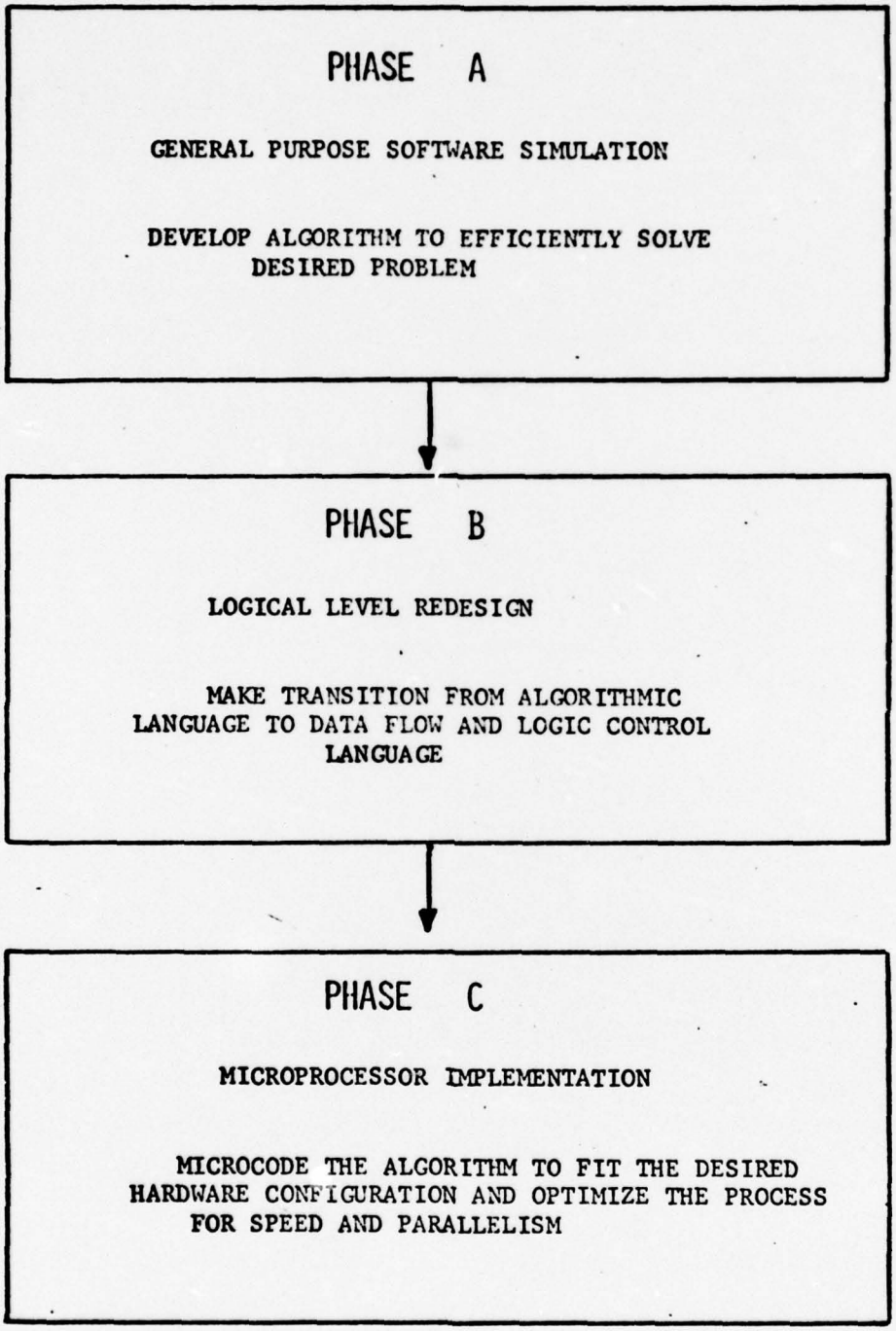


FIGURE 1-1. STEPS OF THREE-PHASE PROGRAM

microcode does not result in a high performance realization of the algorithm. What is necessary is the reconstruction of the algorithm in terms of its data flow and logical component interconnections.

The following sections of the report describe this reconstruction for the block processing approach to stereo image matching. These results will be used in Phase C of the current effort to actually benchmark the process on an array of microprocessors.

2.0 ALGORITHM REFINEMENTS

In an effort to firm up the block processing algorithm for logic level reconstruction, two refinements were made that were not included in the technical report for Phase A. One involved the addition of a reliability measurement capability, and the other involved the further sophistication of the correlation patch shaping scheme.

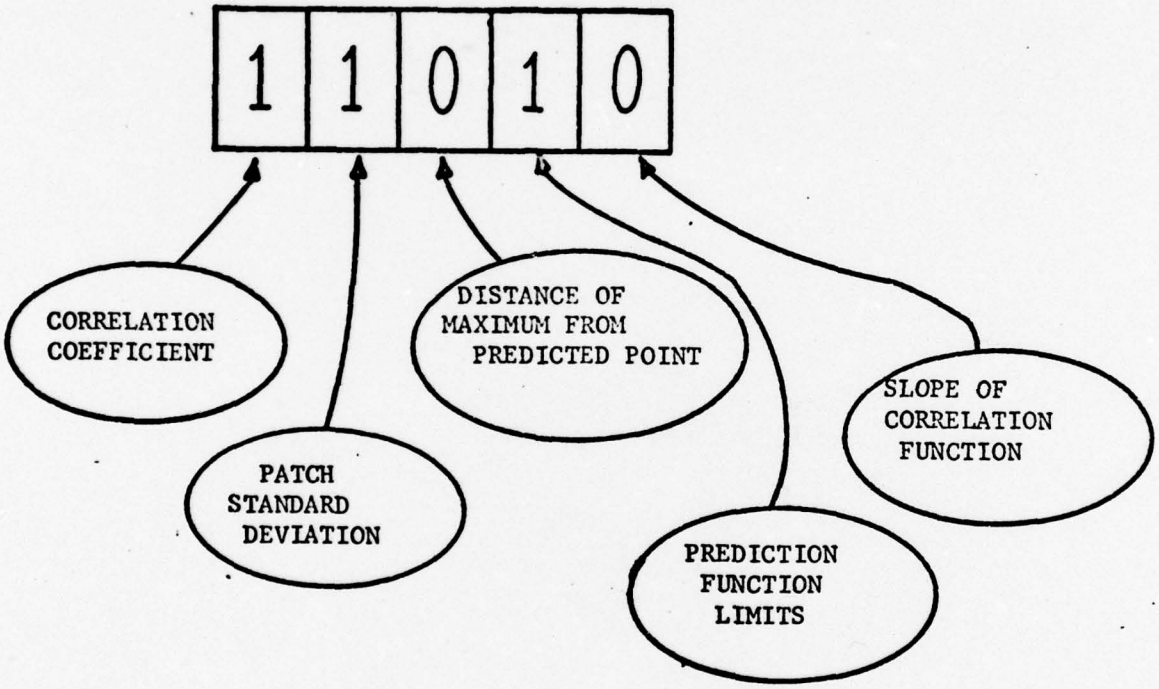
2.1 Match Point Reliability Determination

It is desirable for analysis purposes to obtain for each match point generated by the block process a measure of the point's reliability in terms of internal matching parameters at the time the point is being generated. Therefore, a reliability factor was devised which characterizes the reliability of a match point with respect to the following criteria:

- value of the correlation coefficient
- gray-scale standard deviation of the A image patch
- distance of the correlation maximum from the predicted location
- range limits of the prediction function
- slope of the correlation function

The form of this reliability factor is illustrated in Figure 2-1.

AN N DIGIT NUMBER, ONE DIGIT FOR EACH RELIABILITY CRITERION:



A RELIABLE MATCH POINT HAS A FACTOR OF 0

FIGURE 2-1. MATCH POINT RELIABILITY FACTOR

As is shown, the reliability factor is an N digit number, one digit that is either 1 or 0 for each reliability criterion. In implementation, each criterion value for a match point is tested against a preset cutoff value. If the test fails, a 1 is placed in the factor for that criterion. Therefore, a match-point that is reliable with respect to all the criteria has a reliability factor of zero. At the end of the matching run a reliability summary is printed which includes both the percentage of totally reliable points and the percentage of unreliable points with respect to each criterion.

The reliability factor as described performs a number of useful analysis functions both in the continued modification of the algorithm and in the final evaluation of match points. If the analyst working on the algorithm discovers a new technique that he feels is theoretically effective for a section of the process, he may observe the effect of this technique directly in the reliability summary after he has made the change in the algorithm, all other things being the same in the matching run. As an example, the patch shaping algorithm described in the next section was thought to be more accurate than the algorithm that was currently implemented. The new algorithm was inserted in the block matching code and the result was that the number of totally reliable match points increased from 78% to 96%, verifying that for that particular data the new algorithm was clearly superior.

As an internal aid to the matching process the reliability factor for each match point can provide the information necessary for rematching certain points. This falls under the matching philosophy that an attempt be made to improve unreliable points as they are generated, when the process has at its disposal all that is to be known about the points, rather than allowing the process to roar through the data, leaving unreliable points to be resolved by a postprocess.

In rematching, the individual digits of the reliability factor act as a decision table. For instance, if on a given point the value of the correlation coefficient is low and the standard deviation is low, then rematching of the point should occur with a larger patch size. When the rematch is complete, the new reliability factor can be compared with the old to determine if improvement has occurred. In another situation, if the value of the correlation coefficient is low and if the distance of the correlation maximum from the predicted maximum is great or if the prediction function has exceeded its limits, then rematching should occur around a new predicted point and consequently with

a new patch shaping. These are but two of the strategies that can be developed through use of the reliability factor.

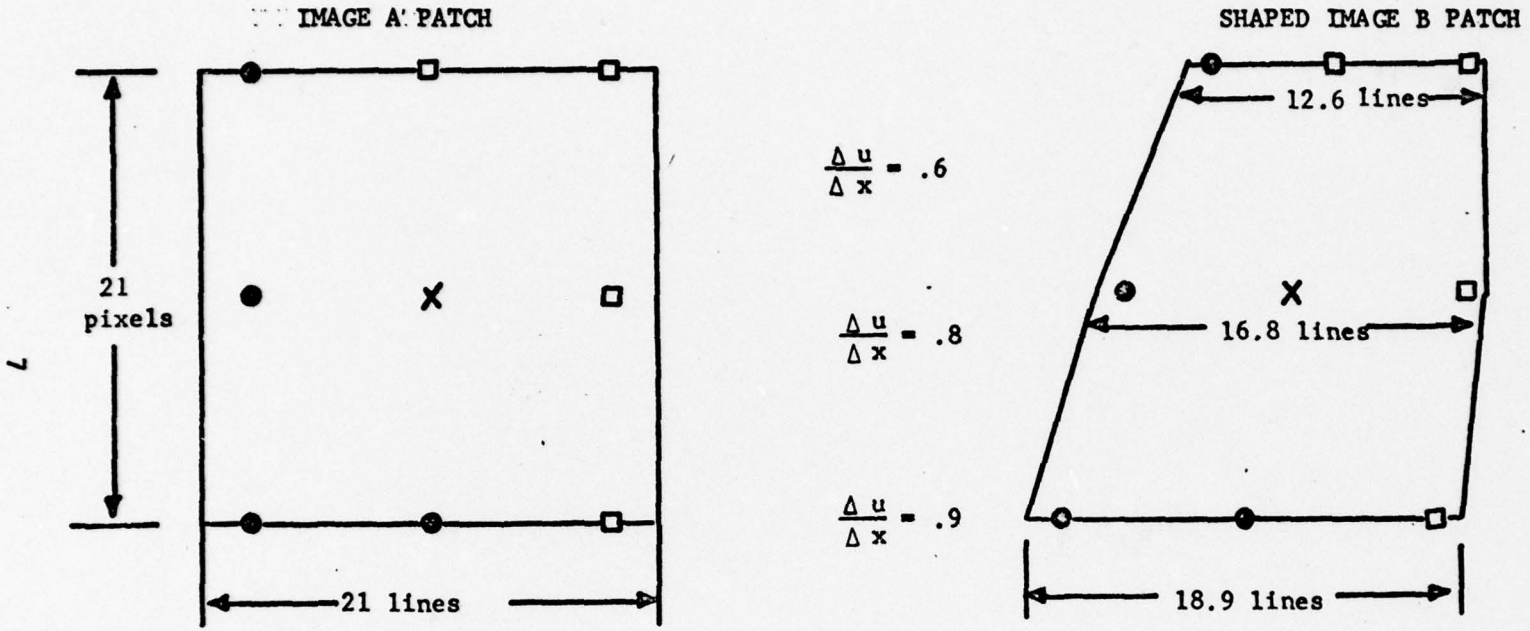
As an external aid the reliability factor can be used as a weighting coefficient in all subsequent processes that make use of match points.

2.2 Improved Correlation Patch Shaping

During the block matching algorithm development of Phase A, correlation patch shaping was implemented as a compression or expansion of the B image patch sides in the parallax dimension. The shape function was constant over the entire patch and consequently the patch was always either square or rectangular. This scheme did produce satisfactory matching results but sometimes with rather low values of the correlation coefficient.

In an attempt to more closely model the terrain during matching and to provide more representative values of the correlation coefficient, the patch shaping scheme was modified as follows. Referring to Figure 2-2, it can be seen that over the somewhat standard 21 x 21 pixel patch area there occur 9 match positions. At any given patch placement for matching, 4 of these positions have actually been matched, 4 have been extrapolated from the known matches, and one is the predicted position for the current match. These 9 positions are evenly spaced on the A image patch, but they serve as inflection points for shaping the B image patch. As is shown in the figure, both the rate of change prediction function $\Delta u/\Delta x$ and the scan line starting position can vary for each of the three rows of three points. These differentials have been taken into account in shaping the B image patch under the new scheme. As a result, the B image patch is more nearly a six-sided figure and has more degrees of freedom in conforming to the terrain. The inclusion of the new shaping scheme did produce higher values of the correlation coefficient in widely varying terrain.

Figure 2-2. Correlation Patch Shaping



● = PREVIOUSLY MATCHED POINT

X = POINT TO BE MATCHED NEXT

□ = EXTRAPOLATED POSITION OF FUTURE MATCH POINT

3.0 BLOCK PROCESS LOGICAL CONSTRUCTION

The block matching algorithm has been analyzed and divided into its discrete logical components. The criterion for constructing these components and for assigning names to them is purely functional. Each component performs one logical function in the process. Some are larger than others in terms of number of coded statements, and some require more execution time than others. The current concepts of top-down software organization have been used throughout such that each logical component has but one entrance and one exit in terms of control structure and such that each component operates on one set of data or parameters.

3.1 Process Organization

The interconnections of the logical components are shown in Figure 3-1. The top-down organization has been rotated into a left-to-right organization for presentation purposes. Following is a functional description of each logical component.

BMATCH - Basic process control loop; one time through the loop produces one match point.

SETUP - Initializes the image data buffer pointers and parameters; establishes interface to input units; called once only.

PARAMS - Initializes relative orientation and matching parameters; accepts starting match points; called once only.

PREDICT - Responsible for predicting the next point to be matched on both the A and B images; makes use of epipolar geometry and current rate of change functions; dependent on UPDATE for valid B image predictive data.

ECORL - Computes a correlation coefficient for each site on the epipolar line passing through the predicted point; accumulates all necessary sums, sums of squares, and cross products of image gray-scale data.

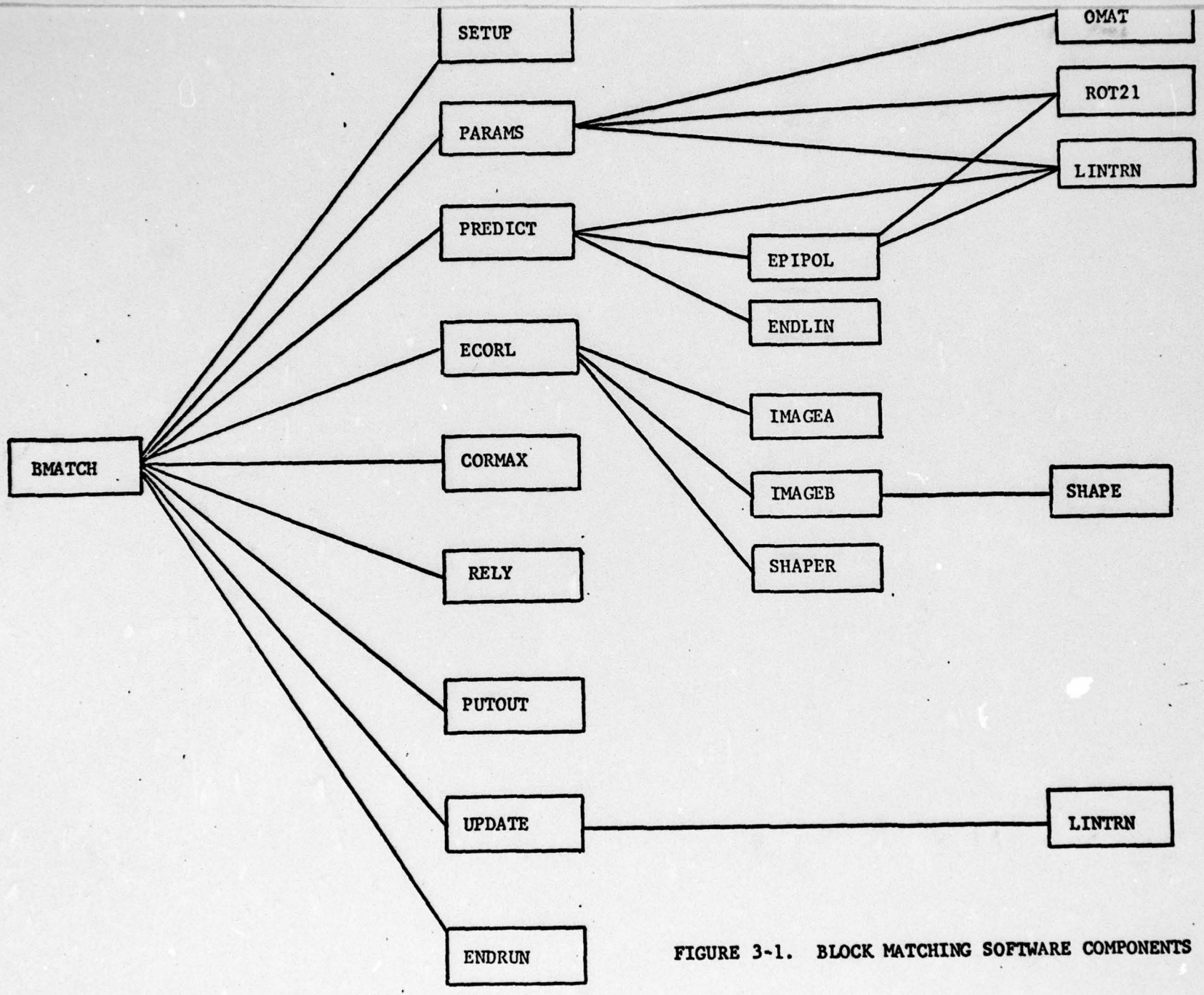


FIGURE 3-1. BLOCK MATCHING SOFTWARE COMPONENTS

CORMAX - Determines the search site of maximum correlation and interpolates the maximum to a fraction of a pixel; determines slope of the correlation function.

RELY - Determines the reliability of a match point based on preset cut-off values; computed reliability factor is based on value of the correlation coefficient, standard deviation over the correlation subregion, slope of the correlation function, and distance of the correlation maximum from the predicted point.

PUTOUT - Responsible for interfacing to output units; formats current matching parameters for display on printer; formats match point for recording on mag tape or disk.

UPDATE - Uses current match point information to update the prediction mechanism; keeps track of last digital position on B image and refines B image rate of change functions.

ENDRUN - Generates a printed reliability summary based on all points matched; called once only at end of matching run.

IMAGEA - Responsible for making A image input data available to ECORL; checks for buffer irregularities during the process.

IMAGEB - Responsible for making B image input data available to SHAPER; checks for buffer irregularities during the process.

SHAPE - Computes B image patch shaping parameters based on current size and rate of change functions in the vicinity of the patch.

SHAPER - Generates one row of shaped B image data for use by ECORL; uses parameters computed by SHAPE; called once for every pixel row of correlation patch.

EPIPOL - Computes the coefficients of corresponding epipolar lines on the A and B images passing through the predicted point; coefficients used by PREDICT.

- **ENDLIN** - Checks for wandering blocks in a complete line of match points (all points with the same x coordinate on the A image); corrects any blocks found wandering; called only when a line of match points is complete.

OMAT - Computes the 3 x 3 orientation matrix from the exposure tilt angles ω ϕ, K; called once only by PARAMS.

ROT21 - Performs the rotation of a point in one coordinate system into another coordinate system on the basis of a 3 x 3 orientation matrix.

LINTRN - Performs the linear transformation of a point from one two-dimensional coordinate system into another; used for transforming digital scan coordinates to photo coordinates and vice versa.

As a further explanation of the process, BMATCH contains only a control loop that calls the necessary components into operation in the proper sequence. As will be seen later, BMATCH simulates the operation of a host computer in sequencing the operation of other processors. The actual component sequence is shown in Figure 3-2. It is set up such that one execution of the control loop produces one match point.

3.2 Data Organization

Just as the matching algorithm in terms of process control has been organized into logical components, so too the data and parameters operated on by the algorithm have been organized into discrete packets. These packets are used as communication links between the various process components. Figure 3-3 shows this data communication and Figure 3-4 is a table showing all of the variables in each packet and the occurance of each packet over the collection of logical components. A star indicates that the indicated component has a need for the indicated packet. Following is a description of what each packet contains.

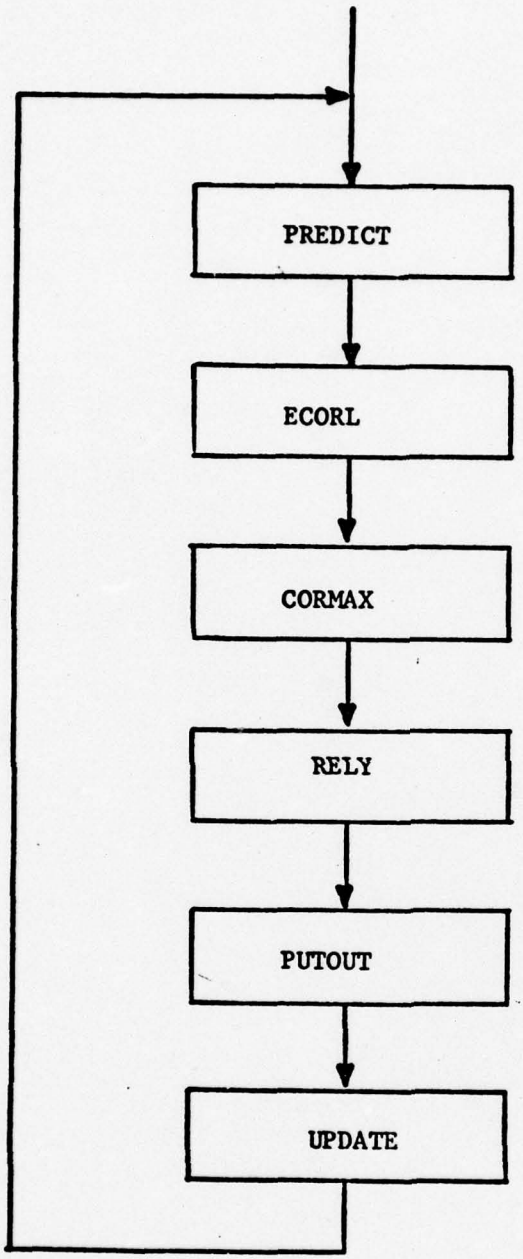


Figure 3-2. Primary Matching Loop (BMATCH)

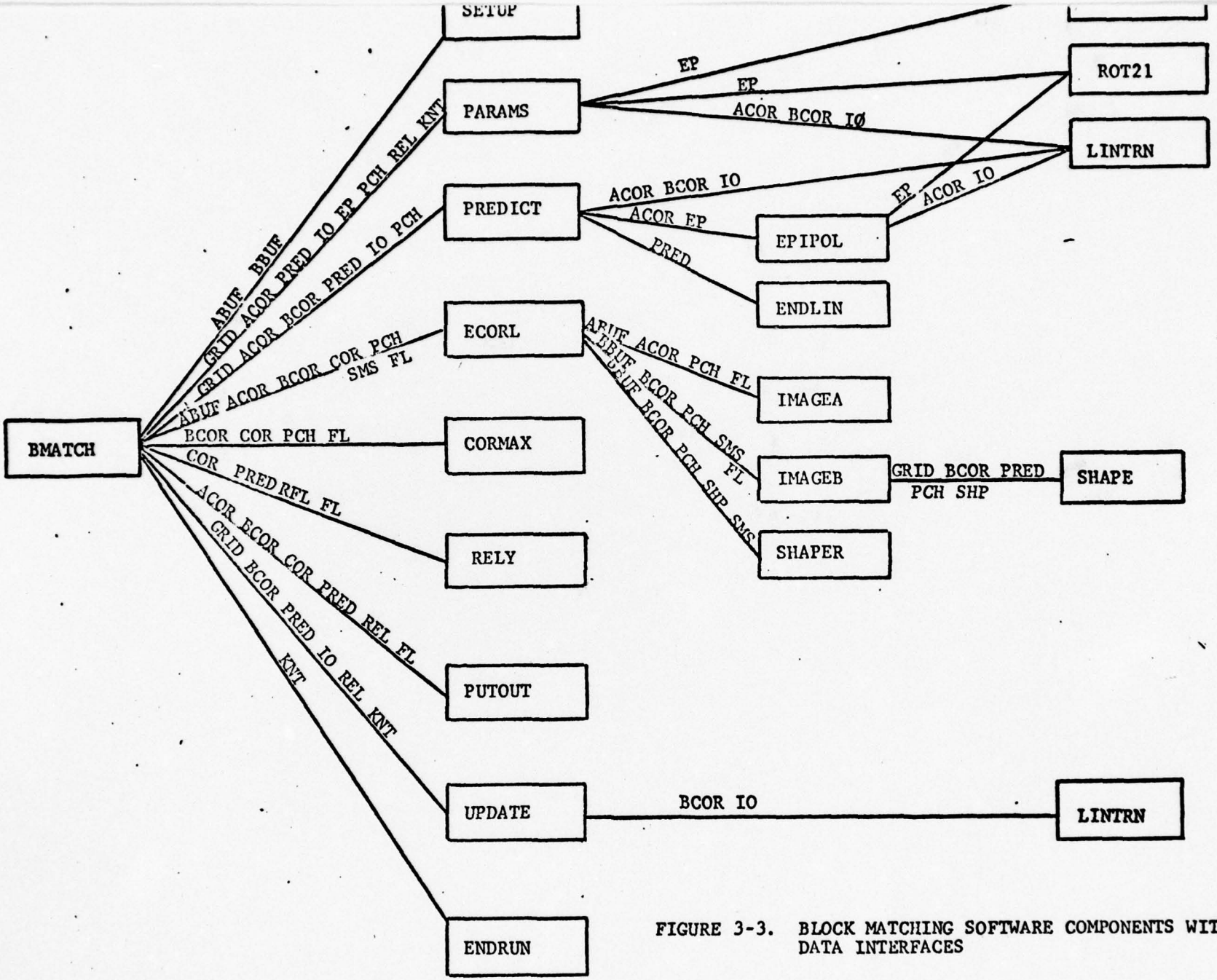


FIGURE 3-3. BLOCK MATCHING SOFTWARE COMPONENTS WITH DATA INTERFACES

	GRID	ABUF	ACOR	BBUF	BCOR	COR	PRED	IO	EP	PCH	SHP	SMS	REL	KNT	FL
	PSTI PNDI PDX PSTJ PNDJ PDY	IMA() IALEN NWA JALEN ILA ILNA JPA JPNA IABP	PA() DPA() IAST IWA ICHA NA MA	IMB() IBLEN NWB JBLEN ILB ILNB JPB JPNB IBBP	PB() DPB() DPBR IBST IWB ICHB NB MB	CORC() LS SDA CPK CSL UDEV	PBR() PBRD() PDU() KP KPN	FXA() FYA() FXIA() FYIA() FXB() FYB() FXIB() FYIB()	BASEB() ORMBA() EA EB	HXA HYA SX SHF	ISH LST LND SH() DSH() RK() DRK() IRK INDX	IPB() SUMB() SSQB() SXY()	TCC TSD TDU TSL IRV() IRFAC	NPTS IRTAB()	IFLA IFLB IFLC
BMATCH	*		*												
PARAMS	*		*				*	*	*	*			*	*	
SETUP		*		*											
PREDICT	*		*		*		*	*	*						
ENDLIN							*								
ECORL		*	*		*	*				*		*			*
SHAPER				*	*					*	*	*			
CORMAX					*	*				*					*
RELY						*	*						*		*
UPDATE	*				*		*	*					*	*	
IMAGEA		*	*							*					*
IMAGEB				*	*					*	*	*			*
SHAPE	*				*		*			*	*				
PUTOUT			*		*	*	*						*		*
EPIPOL			*						*						
ENDRUN														*	

FIGURE 3-4. BLOCK MATCHING DATA INTERFACES

GRID - Starting and ending positions and increments for the matching grid over the desired area of processing.

ABUF - Image A gray-scale data and buffer management parameters.

ACOR - Match point coordinates on image A in digital scan system and photo system and associated parameters.

BBUF - Image B gray-scale data and buffer management parameters.

BCOR - Match point coordinates on image B in digital scan system and photo system and associated parameters.

COR - Correlation search area parameters; values of correlation coefficient, standard deviation, correlation slope, and deviation from predicted point.

PRED - Rate of change prediction functions for each path of blocks.

IO - Interior orientation transformations in X and Y for image A and B to convert digital scan coordinates to data coordinates and vice versa.

EP - Relative orientation parameters and epipolar line coefficients.

PCH - Correlation patch size and shape and search segment length parameters.

SHP - B image correlation patch shaping functions and differential increments.

SMS - Correlation coefficient sums and cross product accumulators.

RFL - Reliability factor and reliability criteria cut off values.

KNT - Reliability frequency accumulators.

FL - Process error flags.

The attempt is being made to maintain the integrity of these data packets as well as the integrity of each logical component as the process is reconstructed in microcode. The reason is that this type of modularization is the key to straightforward management and subsequent modification of the software structure.

4.0 BLOCK PROCESS RECONSTRUCTION

The actual benchmark of the block matching algorithm will take place on a configuration of microprogrammable processors called the Flexible Processor (FP) Array. This is an experimental configuration being constructed by the Digital Image Systems Division for image processing research and demonstration purposes. The configuration contains four Flexible Processors with their associated memory modules controlled by a CDC 1700 computer. This configuration is shown in Figure 4-1. Also shown as part of the configuration is a data channel controller (DCC) with 32K of memory. This is called a line buffer memory and is used to interface image data at very high rates of speed to and from FPs and peripheral devices. A fifth Flexible Processor is included in the configuration as part of a digital scan converter. The role of this FP is to manage the large display memory and to constantly refresh a 512 by 640 element color CRT.

The Flexible Processors in this configuration are all uniform in design and connected together in a pipeline fashion through the standard AQ and DSA channels. In fact, the attempt is being made to construct the entire configuration in what can be considered a standardized fashion. This is the key to being able to implement many different algorithms and applications on the same hardware configuration. This represents a different approach from the FP configurations that have been constructed in the past. In such systems, one of which is the Four Channel Modular Change Detection System, the Flexible Processors in the processing pipeline can be different from one another, having different amounts of micromemory and different communication paths. In short, the configuration is tailored to meet the real-time needs of a truly special purpose processing task. It is difficult to implement a variety of algorithms on such a configuration.

Flexible Processors as well as banks of MOS memory are all designed as modular hardware building blocks. As such large numbers of them can be cabled together to produce a processor of considerable computational power. If the

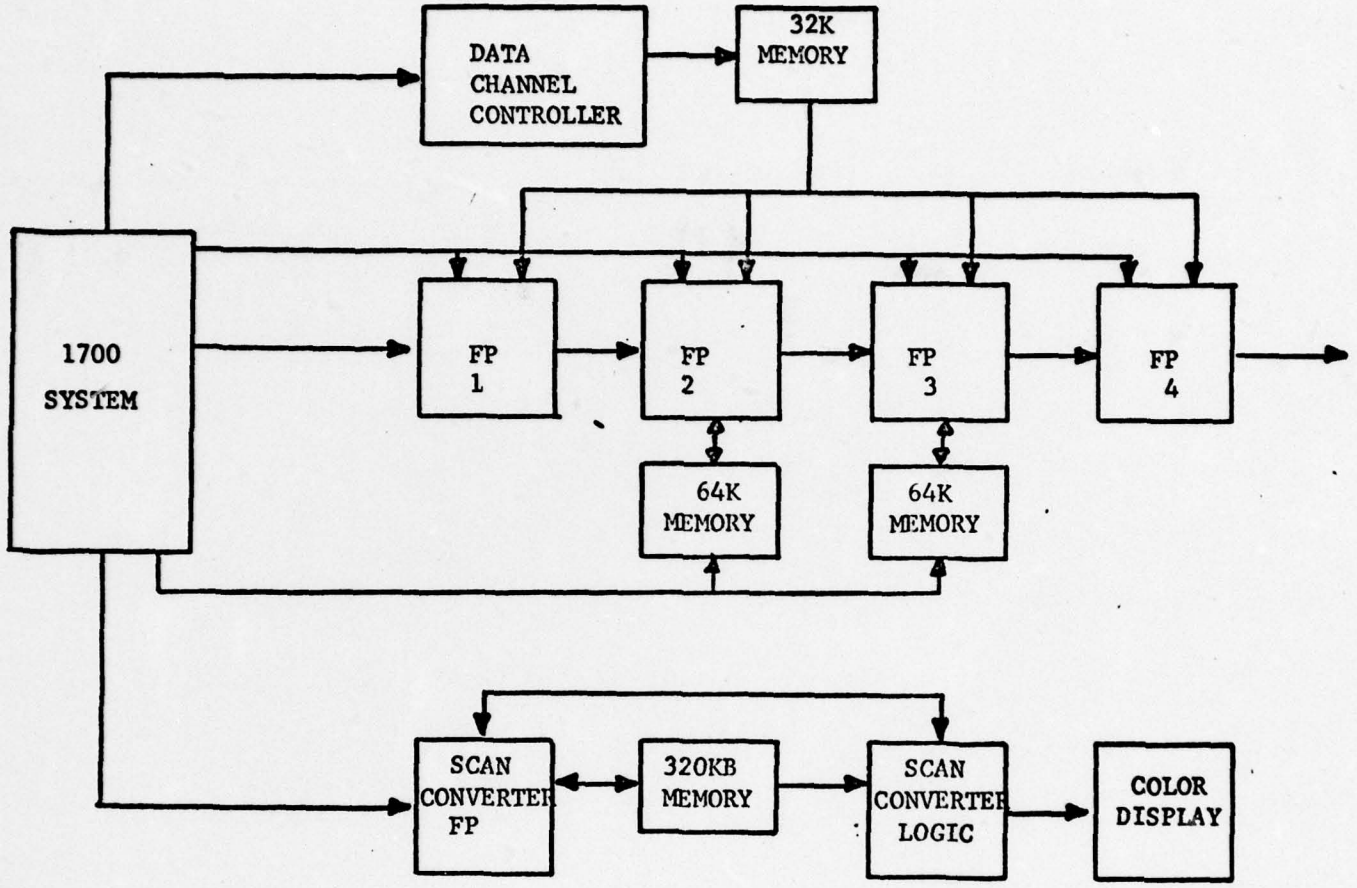


Figure 4-1. FP Array Configuration

construction of the FPs and memory is uniform and the cabling between them standardized, then the resulting configuration is also very flexible as well as expandable.

Figure 4-2 illustrates this concept of expandability. Typically, an algorithm is implemented on a collection of FPs that are connected in a pipeline fashion such that the first FP receives some data and does some computations on it and then passes the results to the next FP along the pipeline. While the second FP is doing its portion of the processing and passing results to the next, the first is processing new data, and so on through the pipe. In this way, processing parallelism is achieved because each FP performs its portion of the algorithm simultaneously with the others.

Such a pipeline of FPs can be considered as one processing channel. To provide more computational power or higher data throughput, more FPs can be added to the pipeline up to a total of 16 per channel. Also as another dimension to the configuration expandability, a channel may be duplicated up to 16 times in the direction shown in the figure. Particularly in image processing, this channel duplication can result in much increased system throughput because the image data can be partitioned such that each channel runs the same algorithm in parallel on its portion of the data. Thus, a four channel configuration can theoretically process the same data as a one channel system in one fourth the time.

However, duplicate hardware channels need not run the same algorithm. Some channels can be loaded with microcode to perform completely independent tasks on different sets of data. In this sense a multiple channel FP configuration can be truly classed as a multiprocessing system.

The above discussion has been provided as background to show that the current benchmark is being performed as a demonstration of capability of a hardware subset that can evolve into a flexible and powerful system.

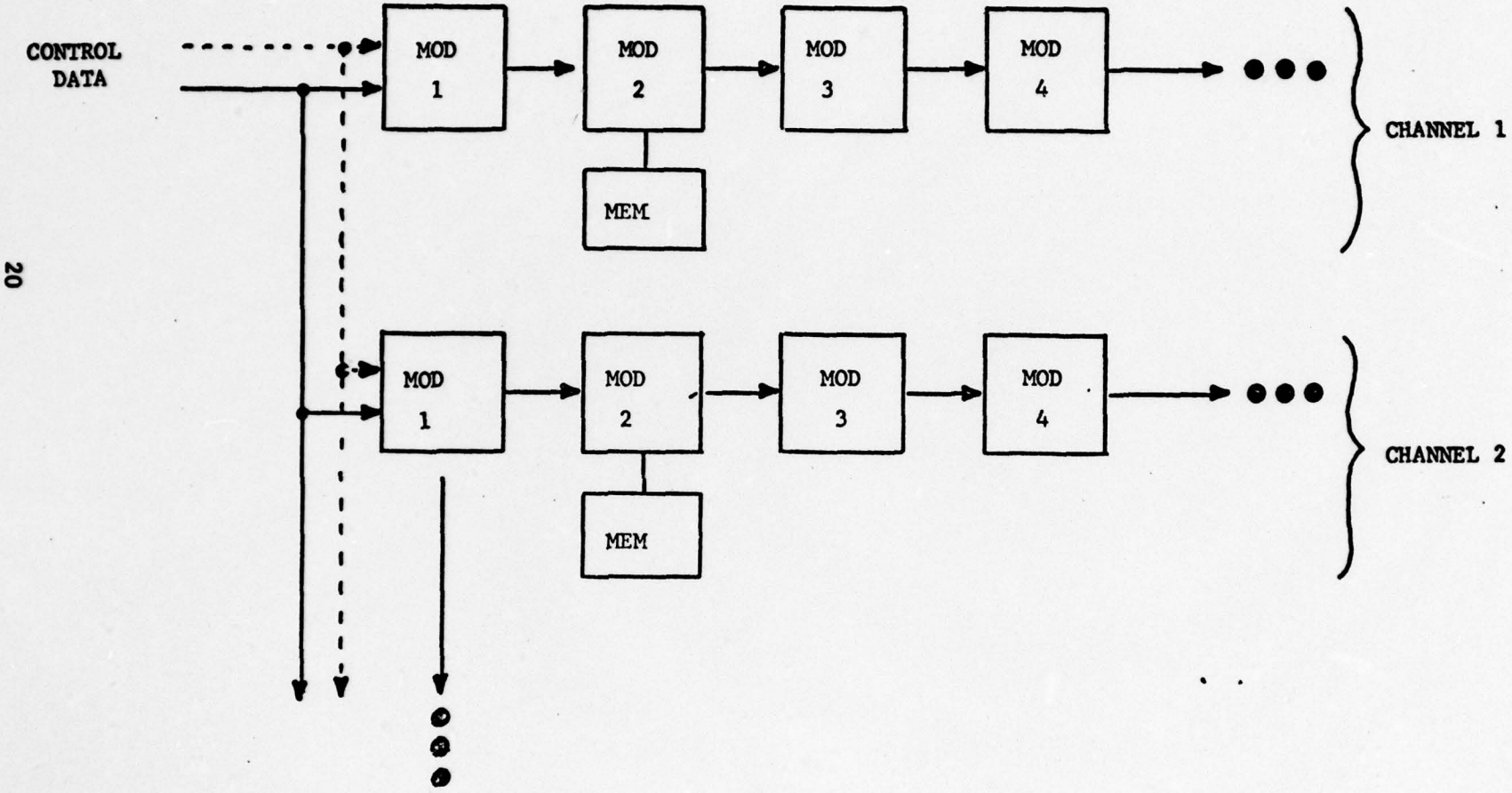


Figure 4-2. Configuration Expandability

4.1 Algorithm Distribution

What is available now to run the benchmark is a configuration of four Flexible Processors arranged in one channel. In the logic level reconstruction that follows Flexible Processors 1, 2, 3, and 4 of Figure 4-1 will be referred as modules 1, 2, 3, and 4 respectively. The CDC 1700 control computer is the logical host. Implementation of the block matching algorithm on the available hardware then involves the distribution of the logical components of the algorithm over the host and four microcode modules.

Figure 4-3 illustrates this distribution. The host module contains the basic process control loop BMATCH and the initialization components SETUP and PARAMS. ENDLIN has been included as part of BMATCH and is not shown because of its minimal function. The components ØMAT, RØT21, and LINTRN have likewise been included as part of their calling components.

The theory of operation of the resulting pipeline is as follows. The host module is responsible for loading the proper microcode, sequencing the process, supplying data to the large MOS memory, and depositing the output match points on disk. MOD1 computes the predicted match point locations for an entire line of blocks and passes this information to MOD2. It then receives refined match point data from MOD4 and starts the prediction for the next line of blocks. Meanwhile MOD2 uses the data received from MOD1 to compute the patch shaping functions and to start accessing image data from memory. As soon as one patch row of data is shaped, it is sent to MOD3 for statistical accumulation and the next patch row is shaped. As this is occuring, MOD3 accumulates the necessary partial sums for the next row. For each row of the patch MOD2 and MOD3 are constantly talking to one another. When MOD3 finishes the last row of the patch, it sends the correlation data to MOD4 which determines the maximum and reliability factor and prepares to output the match point data to the host and MOD1. At this time, MOD2 has gone on to the next block. This goes on in feedback fashion until the host has determined that the process is complete.

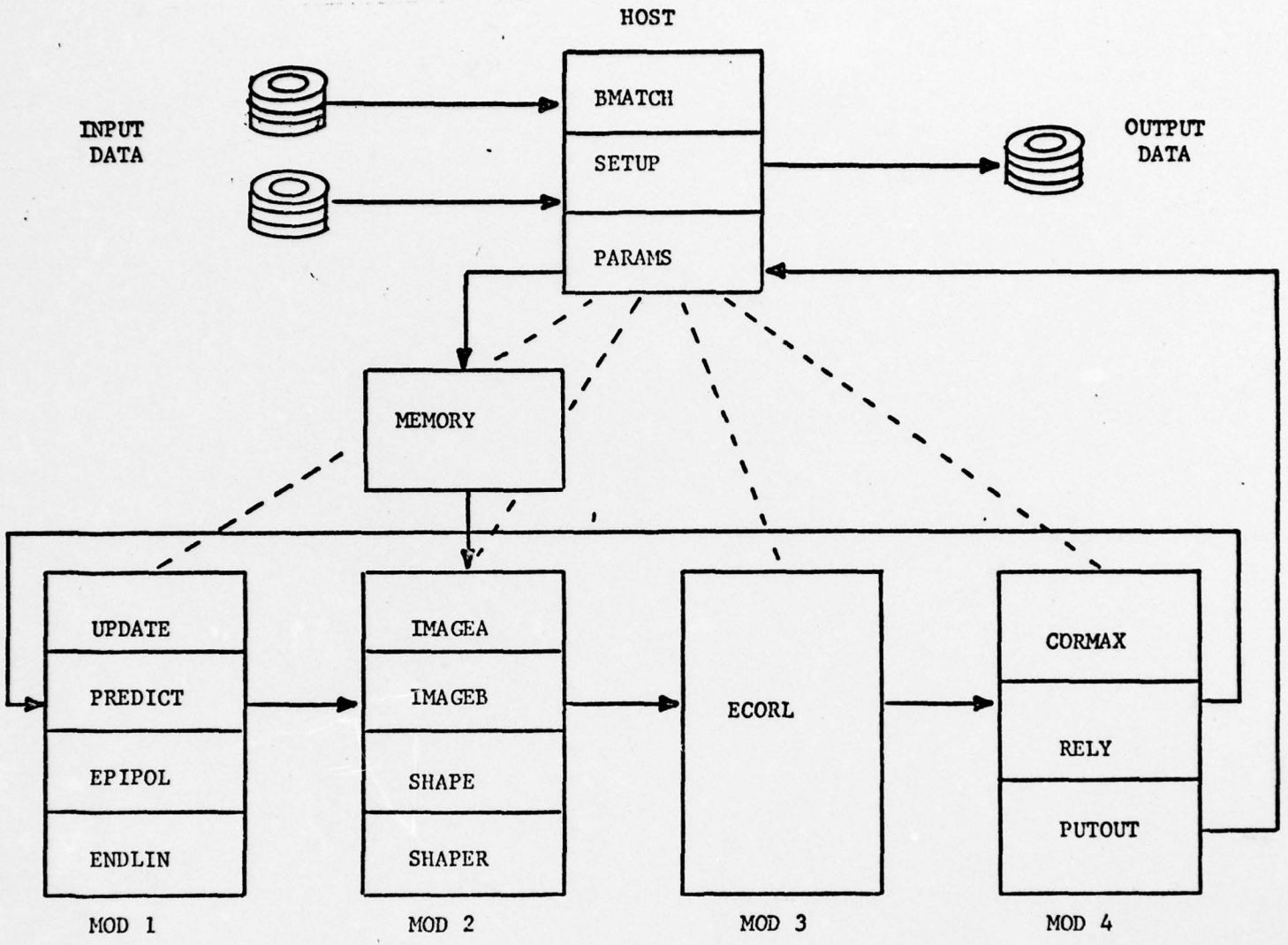


Figure 4-3. Block Matching Microprocessor Implementation

In breaking apart each of the processing modules, it has been found that the relative computational complexity in equivalent add operations per match point is as follows:

MOD1	159
MOD2	5851
MOD3	23874
MOD4	143

It can be seen that MOD3 is the workhorse of the process.

The above represents a rather conservative distribution of the algorithm's compute load. Theoretically, in a distributive computing network such as the FP Array, one should be able to take the total number of operations or the total compute load of the algorithm and divide it by four, thus evenly distributing the load among the four processing modules. The problem is that some components contain many microcode statements to perform a few operations whereas other components contain few microcode statements in tight loops that execute many operations. ECORL is a case of the latter. Each FP has but 1K words of micromemory, although many operations can be performed in one microinstruction. Therefore, the problem of redistributing the above compute load is dependent upon how many logical components can fit in each module's micromemory.

An alternative implementation which incorporates a different compute load distribution is shown in Figure 4-4. The relative computational complexity is as follows:

MOD1	903
MOD2	7812
MOD3	21669
MOD4	143

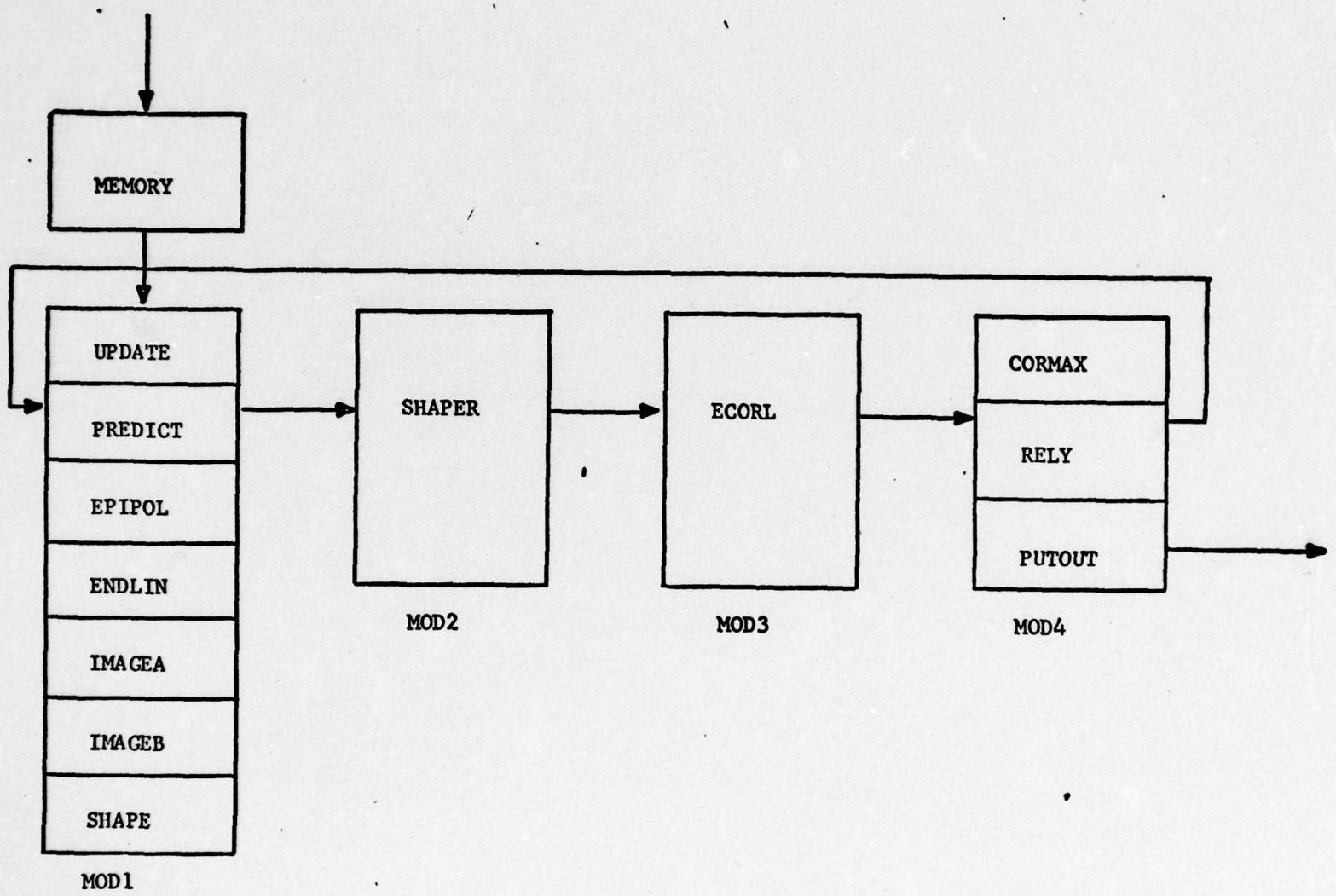


Figure 4-4. Alternate Implementation

Here, some of the responsibility for accumulating gray scale sums has been taken from ECORL in MOD3 and incorporated into SHAPER in MOD2. Also, the image data accessing components and the shaping computations have been pushed back to MOD1. MOD4 has been left relatively unloaded because it must spend a lot of time communicating with the host as regards output data, a slow process in comparison to the extremely fast computations of the other modules.

These implementations have been provided as initial estimates to size up the microcoding task. Further redistribution is possible when the microprogramming is actually performed. One area to be considered is whether some of the responsibility of MOD3 can be placed in MOD4. Another aspect to be firmed up during actual microprogramming is how to obtain the optimum process parallelism such that the time spend by a module waiting for another module is minimized.

5.0 BLOCK PROCESS TIMING ESTIMATE

In determining how fast the benchmark configuration will actually execute the block matching algorithm over a stereo model area, the following points are relevant:

- Matching occurs for approximately 12000 points per square inch of image.
- A typical stereo overlap is 54 square inches.
- Therefore, 648000 points per model area are generated.
- In a general purpose environment, where the block matching algorithm has a rate of 14 points per second, a model area would take approximately 13 hours of processing time.
- In the benchmark configuration, the processing time is dependent on the number of operations per point required by the most heavily loaded processor in the configuration, provided that the optimum parallelism has been achieved.
- The time required by a Flexible Processor to perform one add operation is 125 nanoseconds.

With these considerations in mind, the benchmark timing should fall within the range of the following table, depending on the equivalent number of sequential operations performed by the most heavily loaded processor.

OPERATIONS PER MATCH POINT	SECONDS PER MATCH POINT	TOTAL MODEL TIME IN MINUTES
10000	.00125	13.5
15000	.00188	20.3
20000	.00250	27.0
25000	.00313	33.8
30000	.00375	40.5
35000	.00438	47.3

These times have been derived in accordance with matching data where the image pixel side covers 4 feet on the ground, the correlation patch size is 21 by 21 pixels and the match point grid interval is 10 pixels by 8 lines. This is a somewhat typical matching situation. But the times are subject to change with different image data depending on the scale of the imagery, density of matching and correlation patch and search segment size.

6.0 CONCLUSION

The block matching algorithm has been logically reconstructed and is ready for microprogramming. A number of problems involving the optimum distribution of logical components and the size of the available micromemories have been uncovered in Phase B which should be resolved in Phase C during microprogramming. The attempt during Phase B has been to keep the development on a rather high logical level, deferring discussions of communication and data channels, instruction formats, and I/O transfer rates to the microprogramming task of Phase C.